Convex Non-Negative Matrix Factorization With Adaptive Graph for Unsupervised Feature Selection

Aihong Yuan, Mengbo You, Dongjian He, and Xuelong Li, *Fellow, IEEE*

***Abstract*—Unsupervised feature selection (UFS) aims to remove the redundant information and select the most representative feature subset from the original data, so it occupies a core position for high-dimensional data preprocessing. Many proposed approaches use self-expression to explore the correlation between the data samples or use pseudolabel matrix learning to learn the mapping between the data and labels. Furthermore, the existing methods have tried to add constraints to either of these two modules to reduce the redundancy, but no prior literature embeds them into a joint model to select the most representative features by the computed top ranking scores. To address the aforementioned issue, this article presents a novel UFS method via a convex non-negative matrix factorization with an adaptive graph constraint (CNAFS). Through convex matrix factorization with adaptive graph constraint, it can dig up the correlation between the data and keep the local manifold structure of the data. To our knowledge, it is the first work that integrates pseudo label matrix learning into the self-expression module and optimizes them simultaneously for the UFS solution. Besides, two different manifold regularizations are constructed for the pseudolabel matrix and the encoding matrix to keep the local geometrical structure. Eventually, extensive experiments on the benchmark datasets are conducted to prove the effectiveness of our method. The source code is available at: https://github.com/misteru/CNAFS.**

***Index Terms*—Adaptive graph constraint, manifold structure, non-negative matrix factorization (NMF), unsupervised feature selection (UFS).**

Manuscript received June 28, 2020; revised October 4, 2020; accepted October 26, 2020. This work was supported in part by the Natural Science Foundation of Shaanxi Province under Grant 2020JQ-279, and in part by the Doctoral Start-Up Foundation of Northwest A&F University under Grant Z1090219095 and Grant Z109021803. This article was recommended by Associate Editor S. Cruces. *(Aihong Yuan and Mengbo You are co-first authors.) (Corresponding author: Mengbo You.)*

Aihong Yuan and Mengbo You are with the College of Information Engineering, Northwest A&F University, Xianyang 712100, China, also with the Key Laboratory of Agricultural Internet of Things, Ministry of Agriculture and Rural Affairs, Northwest A&F University, Xianyang 712100, China, and also with the Shaanxi Key Laboratory of Agricultural Information Perception and Intelligent Service, Northwest A&F University, Xianyang 712100, China (e-mail: ymb@nwafu.edu.cn).

Dongjian He is with the College of Mechanical and Electronic Engineering, Northwest A&F University, Xianyang 712100, China, also with the Key Laboratory of Agricultural Internet of Things, Ministry of Agriculture and Rural Affairs, Northwest A&F University, Xianyang 712100, China, and also with the Shaanxi Key Laboratory of Agricultural Information Perception and Intelligent Service, Northwest A&F University, Xianyang 712100, China.

Xuelong Li is with the School of Computer Science and Center for Optical Imagery Analysis and Learning, Northwestern Polytechnical University, Xi'an 710072, China.

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2020.3034462.

Digital Object Identifier 10.1109/TCYB.2020.3034462

I. Introduction

In THE field of computer vision and multimedia data analysis, most of the original data have a very high dimensionality, which makes the data analysis very difficult due to the high calculation and storage costs [1]–[3]. In practical applications, lots of features are relevant or redundant [4]–[6]. Therefore, data analysis would be more effective if the representative and important features are selected to represent the raw data. Feature selection (FS) is such a data preprocessing technique and it has received considerable critical attention in recent years [7]–[10].

FS aims to remove the redundant and irrelevant features and choose a subset representative and important features through certain criteria [11]. Recently, there has been renewed interest in FS for high-dimensional data preprocessing in many tasks [12]–[14], such as face recognition [15], [16]; clustering [17]–[19]; etc. In terms of whether or not labels are used during selecting features, FS methods can be categorized into three types, that is: 1) supervised methods [20]; 2) semisupervised methods [21], [22]; and 3) unsupervised methods. In practice, obtaining labels is time-consuming and laborious work, so unsupervised FS (UFS) has attracted the tremendous attention of many researchers and achieved a lot of inspiring results in the past decades.

So far, UFS methods can be roughly categorized into three types: 1) filter methods; 2) wrapper methods; and 3) embedding-based methods. The filter methods use the statistical properties of features to select representative features without any learning algorithm [23]–[25]. Hence, the filter methods are independent of the classifiers and they have low computational cost, while the performance of these methods is not quite satisfied. The wrapper methods usually use the learning algorithm to select the typical features and evaluate the selected features according to the predetermined learning algorithm [26]–[28]. However, there are certain drawbacks associated with the use of the learning algorithm. One of these is that FS performance relies on the quality of data for learning. It also needs a large amount of calculation, especially on large-scale datasets. The embedding-based methods transform the FS into an optimization problem [29], in which the embedding subspace and FS matrix are learned simultaneously. Compared with the former two types of methods, it was reported that the embedding-based methods were able to sort features by importance with low computational cost [30]–[32].

A. Motivation and Overview

Recent advances in embedding-based methods have facilitated the investigation of the self-expression model and pseudolabel matrix learning. Specifically, the self-expression model explores the correlation between data samples, while pseudolabel matrix learning attempts to learn the real label of the data matrix and it transforms the UFS problem into a "supervised learning" task. Moreover, the pseudolabel matrix can also be seen as the low-dimensional manifold of the original data. Both of the two modules (i.e., self-expression module and pseudolabel matrix learning module) can improve the performance of the FS method. Therefore, it is necessary to embed the self-expression module and pseudolabel matrix learning module into the FS method. However, there is currently no literature demonstrating these two modules assembled together for FS. That is, mainly because it is difficult to design a model to combine the self-expression module and pseudolabel matrix learning module together. Specifically, simply adding two modules together does not take the interaction between the two modules into account. In other words, the parameters in the self-expression module do not update while learning the pseudolabel matrix. Conversely, the pseudolabel matrix learning is also not constrained by iteratively optimizing the self-expression module, which is similar to applying either module merely or applying two modules independently. (See Section III-B4 for detailed analysis.) Therefore, how to dynamically combine the self-expression and pseudolabel matrix learning into a single optimization process is vital to improve FS performance.

Motivated by the aforementioned two reasons, we present a novel UFS method via convex non-negative matrix factorization (NMF) with an adaptive graph constraint (CNAFS). Specifically, geometrical manifold learning is utilized in the proposed CNAFS to perform the local structure learning and FS simultaneously. Furthermore, self-expression and pseudolabel matrix learning are cleverly united via two manifold regularization terms. The reason is that the coding matrix in self-expression and the pseudolabel matrix can be seen as two different manifolds of the original data, and we should make sure the local structure information of the original data can be kept in the learned low-dimensional manifold space. In addition, convex NMF (CNMF) is used to implement data self-expression. We choose CNMF as a self-expression scheme for the following considerations: 1) CNMF uses the original data to construct the basis vector space that is consistent with the idea of self-expression; 2) NMF is a partial-based data representation method that increases the interpretability of the corresponding data, and CNMF is one of the NMF methods; and 3) CNMF is not only suitable for non-negative data but also for general data, so the proposed CNAFS with CNMF can work on a variety of data.

B. Contributions

The key contributions can be summarized as follows.

1) Self-expression and pseudolabel matrix learning are cleverly united and to our knowledge, it is the first work that applies the two modules simultaneously to address the UFS problem. Self-expression explores the internal structure of the original data and the pseudolabel matrix learning explores the mapping relations between the data and the labels.
2) Two different manifold regularizations are constructed for the pseudolabel matrix and the encoding matrix to keep local geometrical structure of the original data. Different from existing methods that obtain the similarity matrix by predefined handcraft functions, we compute the similarity matrix by an iteratively update strategy to gain better representation of similarity among features. According to the manifold regularization, the proposed method is supposed to learn a more effective feature selecting matrix.
3) This article proposes an alternative iterative algorithm to solve the five variables involved in the proposed model. Complete theoretical proof procedures for the convergence of the proposed model are given in detail and the computational complexity is also analyzed. Besides, extensive experiments on eight benchmark datasets are implemented to verify the effectiveness of the proposed method, and ablation study is used to show the effect of each module of the proposed model. All the experimental results prove that the proposed CNAFS achieves the best performance among the current state-of-the-art UFS methods.

C. Organization

The remainder of this article is organized as follows. In Section II, some related works for UFS are briefly introduced. The detail of our method for UFS is carefully presented in Section III. To validate the proposed method, the experimental results are shown and analyzed in Section IV. Finally, Section V makes a brief conclusion for this article.

II. Related Work

In this section, we briefly introduce the methods of FS. As mentioned in Section I, the UFS methods can be roughly divided into three categories: 1) filter methods; 2) wrapper methods; and 3) embedding-based methods.

A. Filter Methods

The filter methods utilize statistics to "filter" the original features. The importance of each feature is distinguished by using statistical properties. Therefore, the filter methods are statistical-based methods and without any learning algorithm. Variance score [23] used variance to measure the importance of each feature and the feature with large variance will be selected. Fisher score [24] used the Fisher criterion to select the features. He *et al.* [25] proposed an FS method with Laplacian score (LS). For each feature, the LS is calculated to evaluate the locality preserving power and the importance of the feature. LS models the local geometric structure and seeks features that correlate the graph structure. mRMR [33] was proposed by Peng *et al.*, which was an FS method

based on mutual information. Specifically, a series of intuitive measures of redundancy and relevance is employed to select appropriate features. Masaeli *et al.* [34] proposed a converting transformation-based method via l_1/l_∞ regularization. This method converted linear discriminant analysis (LDA) and Hilbert-Schmidt independence criterion (HSIC) to two new FS algorithms. Tabakhi *et al.* [35] proposed a UFS method based on ant colony optimization. This method used the similarity between features to compute the feature relevance. Although the filter methods have yielded some results, such methods do not select the most important features.

B. Wrapper Methods

Unlike the filter methods which does not consider the follow-up learners, the wrap methods directly take the performance of the learners as evaluation criteria of the feature subset. In other words, the purpose of wrapper methods is to select the feature subset that best fits its performance for a given learner. Under the guidance of this idea, some wrapper methods have been proposed, which combined the FS with unsupervised learning tasks together. Dy and Brodley [36] explored the UFS problem through feature subset selection using expectation-maximization clustering and two different performance criterions. Wolf and Shashua [26] used least-squares optimization process and exploited spectral properties of the candidate feature subsets to guide the FS. Maldonado and Weber [27] proposed a novel FS method based on support vector machines (SVMs) with kernel function. This method was based on sequential backward selection and used the errors in a validation subset as the measure to decide which feature needs to be removed in each iteration. Bermejo *et al.* [28] embedded the classifier into the wrapper algorithm to speed up the FS process. Since the wrapper methods try to find the most suitable features for the learning tasks, these methods are able to achieve a better performance than the filter methods. However, the wrapper methods need to train the learners on every selected features, so the time cost is expensive in FS process.

C. Embedding-Based Methods

The embedding-based methods have drawn many attentions in recent years. These methods incorporate FS into a part of model construction. Specially, the feature subset is automatically selected during the model training. Many typical embedding-based methods use the spectral analysis and sparse constraint for feature subsect selection. Zhao and Liu [37] proposed a unified framework for supervised and unsupervised FS-based spectral analysis. This method employed the spectrum of the graph to measure the correlation between features. Multicluster FS (MCFS) [38] was proposed for UFS via manifold learning and l_1 regularized models. MCFS tried to select those features so that the multicluster structure information of the original data can be preserved. To exploit the discriminative information, unsupervised discriminative FS (UDFS) [39] was proposed. UDFS incorporated discriminative analysis and $l_{2,1}$-norm regularization into a unified framework. Qian and Zhai [40] proposed a robust UFS (RUFS) to deal with the outliers or noise in the original data. This method utilized NMF and $l_{2,1}$-norm regularization to learn the cluster indicator matrix and select the features, simultaneously. Zhu *et al.* proposed a robust joint graph sparse coding (RJGSC) [30] to preserve the local structure of the original data. RJGSC simultaneously took the joint sparse regression and subspace learning into account and it conducted FS on the basis space of the data. Cui *et al.* [41] proposed a UFS method based on CNMF, which incorporates NMF and multisubspace structure learning into one united model. Lu *et al.* [31] proposed a UFS method based on self-expression model to catch the relationships between the features. To preserve local structure of the data, manifold learning is embedded in the object function. Li *et al.* [32] proposed a generalized uncorrelated regression with adaptive graph (URAFS) to seek the uncorrelated yet discriminative features by the improved sparse regression model. Overall, the aforementioned embedding-based methods highlight the need for both self-expressing module and the pseudolabel matrix learning module. However, to the best of our knowledge, no literature demonstrates combining these two modules into a united model.

III. Our Method

Previous embedding-based methods, e.g., URAFS [32], usually use the spectral analysis technique to address the FS problem. The core idea of these methods is to learn an indicator matrix to guide the FS process. Through projecting the original data into the pseudolabel space, the unsupervised FS problem is transformed as a supervised problem. In this article, a novel UFS method is proposed, which unifies the self-expression and pseudolabel matrix learning into one model. Specifically, self-expression and spectral analysis are subtly combined to solve the UFS problem. Different from previous self-expression methods, our method uses convex NMF, which constraint each column of the basis matrix to be convex combination of the data points and this constraint is more suitable for the data in the real world. In this section, the method proposed in this article will be introduced in detail. Furthermore, the convergence of the proposed method will be theoretically analyzed.

A. Model Construction

Given the input data matrix $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2 \cdots \mathbf{x}_n] \in \mathbb{R}^{d\times n}$, where $\mathbf{x}_i$ denotes the ith data sample and superscript d represents the dimension of the data. First, $\mathbf{X}$ is projected into pseudolabel space. Then, a cluster indicator matrix is learned to "supervise" the FS process. This statement can be written as the following formula:

$$\begin{aligned} &\min_{\mathbf{W},\mathbf{Y}_p,\mathbf{b}} \left\|\mathbf{X}^T\mathbf{W} + \mathbf{1}\mathbf{b}^T - \mathbf{Y}_p\right\|_F^2 + \lambda\|\mathbf{W}\|_p, \\ &\text{s.t.} \quad \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c \end{aligned} \tag{1}$$

where $\mathbf{W} \in \mathbb{R}^{d\times c}$ is a transformation matrix, $\mathbf{b} \in \mathbb{R}^c$ denotes the bias and $\mathbf{1} \in \mathbb{R}^n$ is an all 1 column vector, and c represents the number of clusters. $\mathbf{Y}_p = [(\mathbf{Y}_p^1)^T\ (\mathbf{Y}_p^2)^T \ldots (\mathbf{Y}_p^n)^T]^T \in$

$\mathbb{R}^{n\times c}$ is the pseudolabel matrix and each element of $\mathbf{Y}_p^i \in \mathbb{R}^c$ denotes the ith data whether belongs to the corresponding class, so the element of $\mathbf{Y}_p^i$ is between 0 and 1. $\|\cdot\|_p$ denotes p-norm. Deriving formula (1) and letting its derivative equal to zero, we can obtain the optimal solution of the variable $\mathbf{b}$ as the following formula:

$$\mathbf{b} = \frac{1}{n}\Big(\mathbf{Y}_p^T - \mathbf{W}^T\mathbf{X}\Big)\mathbf{1}. \tag{2}$$

In addition, to select the most representative features, $l_{2,1}$ norm is used to constrain the projecting matrix $\mathbf{W}$. Therefore, when bringing formula (2) into formula (1), and replacing $\|\cdot\|_p$ with $\|\cdot\|_{2,1}$, problem (1) can be rewritten as follows:

$$\begin{aligned}&\min_{\mathbf{W},\mathbf{Y}_p} \big\|\mathbf{C_n}\big(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\big)\big\|_F^2 + \lambda\|\mathbf{W}\|_{2,1}\\ &\text{s.t.}\quad \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c\end{aligned} \tag{3}$$

where

$$\mathbf{C}_n = \mathbf{I}_n - \frac{1}{n}\mathbf{1}_{n\times n}, \mathbf{1}_{n\times n} = \begin{bmatrix}1 & \dots & 1\\ \vdots & \ddots & \vdots\\ 1 & \dots & 1\end{bmatrix},$$
$$\mathbf{C}_n^T\mathbf{C}_n = \mathbf{C}_n\mathbf{C}_n = \mathbf{C}_n.$$

To improve FS algorithms, many matrix factoring methods have been introduced. Considering the real-world applications, negative elements are often meaningless in practical problems, although it is correct to have negative values in the decomposition results from a computational point of view. In this article, we use convex NMF to guide the FS. The optimal problem can be preliminarily written as follows:

$$\begin{aligned}&\min_{\mathbf{W},\mathbf{Y}_p,\mathbf{G},\mathbf{V}} \|\mathbf{X} - \mathbf{XGV}\|_F^2 + \big\|\mathbf{C_n}\big(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\big)\big\|_F^2 + \lambda\|\mathbf{W}\|_{2,1}\\ &\text{s.t.}\quad \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c, \mathbf{G} \geq \mathbf{0}, \mathbf{V} \geq \mathbf{0}\end{aligned} \tag{4}$$

where $\mathbf{G} \in \mathbb{R}^{n\times k}$ and $\mathbf{V} \in \mathbb{R}^{k\times n}$. We call $\mathbf{U} = \mathbf{XG}$ as the basis matrix and each column of $\mathbf{U}$ can be interpreted as a convex combination of data point. $\mathbf{V}$ is the so-called encoding matrix.

To explore the local geometrical structure of data, which is very important for FS, graph regularization is used in this article. The graph regularization is based on the following assumption: if some two data points are similar, their labels and encoding vectors should also be closed to each other. Based on this assumption, model (4) can be improved by the following graph regularization:

$$\alpha tr\Big(\mathbf{Y}_p^T\mathbf{L}\mathbf{Y}_p\Big) + \gamma tr\big(\mathbf{VLV}^T\big) \tag{5}$$

where $\mathbf{L}$ is the Laplacian matrix, α and γ are hyperparameters to control the two regularization, and $tr(\cdot)$ denotes trace of the matrix. $\mathbf{L}$ is computed as the following formula:

$$\mathbf{L} = \mathbf{D} - \mathbf{S} \tag{6}$$

where $\mathbf{S} = [s_{ij}] \in \mathbb{R}^{n\times n}$ denotes similarity matrix of data $\mathbf{X}$. $\mathbf{D} \in \mathbb{R}^{n\times n}$ is a diagonal matrix and its diagonal elements can be represented as follows:

$$D_{ii} = \frac{1}{2}\sum_{j=1}^{n}\big(s_{ij} + s_{ji}\big), i = 1, 2, \dots, n. \tag{7}$$

In many previous works, the similarity matrix $\mathbf{S}$ is usually constructed by the projecting matrix $\mathbf{W}$ or the Euclidean distance of the data points. However, $\mathbf{S}$ in this article is obtained through optimizing the objective function and the optimization will be introduced in the next section. No matter by which way the similarity matrix $\mathbf{S}$ is computed, $\mathbf{S}$ should satisfy the following constraints: 1) $s_{ij} \geq 0$ and 2) $\sum_{j=1}^{n} s_{ij} = 1$. Through the constraints, each row of $\mathbf{S}$ seems to be a probability distribution. So, information entropy theory is naturally used to optimize the similarity matrix $\mathbf{S}$. The inverse of the information entropy of $\mathbf{S}$ can be expressed as follows:

$$\beta\sum_{i=1}^{n}\sum_{j=1}^{n} s_{ij}\log s_{ij}, s_{ij} \geq 0, \sum_{j=1}^{n} s_{ij} = 1. \tag{8}$$

In order to reduce the correlation among the rows of the matrix $\mathbf{V}$, a regularization for $\mathbf{V}$ is added as follows:

$$\varepsilon tr(\mathbf{V}^T\mathbf{QV}) \tag{9}$$

where ε is a hyperparameter and $\mathbf{Q} = \mathbf{1}_{k\times k} - \mathbf{I}_k$.

Combined with the aforementioned analysis, our full model for FS is formulated as follows:

$$\begin{aligned}\mathcal{L} = &\|\mathbf{X} - \mathbf{XGV}\|_F^2 + \big\|\mathbf{C}_n\big(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\big)\big\|_F^2\\ &+ \lambda\|\mathbf{W}\|_{2,1} + \alpha tr(\mathbf{Y}_p^T\mathbf{L}\mathbf{Y}_p) + \beta\sum_{i,j} s_{ij}\log s_{ij}\\ &+ \gamma tr(\mathbf{VLV}^T) + \varepsilon tr(\mathbf{V}^T\mathbf{QV}),\\ \text{s.t.}\quad &\mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c, \mathbf{G} \geq \mathbf{0}, \mathbf{V} \geq \mathbf{0}, \sum_{j=1}^{n} s_{ij} = 1, \mathbf{S} \geq \mathbf{0}.\end{aligned} \tag{10}$$

To this end, all of the model for FS is constructed, then the optimization will be described in detail in the next section.

B. Model Optimization

Model (10) involves five variables (*i.e.*, $\mathbf{G}$, $\mathbf{V}$, $\mathbf{W}$, $\mathbf{Y}_p$, and $\mathbf{S}$), which need to be solved. For this situation, iterative optimization method is a very suitable choice. In this section, iterative optimization is used to solve the five variables.

1) Fix $\mathbf{W}$, $\mathbf{Y}_p$, and $\mathbf{S}$, and Update $\mathbf{G}$ and $\mathbf{V}$: When $\mathbf{W}$, $\mathbf{Y}_p$, and $\mathbf{S}$ are fixed, model (10) is equal to the following formula:

$$\begin{aligned}\mathcal{L}_1(\mathbf{G}, \mathbf{V}) = &\|\mathbf{X} - \mathbf{XGV}\|_F^2 + \gamma tr(\mathbf{VLV}^T)\\ &+ \varepsilon tr(\mathbf{V}^T\mathbf{QV}), \quad \text{s.t.}\quad \mathbf{G} \geq \mathbf{0}, \mathbf{V} \geq \mathbf{0}.\end{aligned} \tag{11}$$

When $\mathbf{G}$ or $\mathbf{V}$ are fixed, $\mathcal{L}_1(\mathbf{G}, \mathbf{V})$ is convex for the other one. So the model (11) can be solved in an iterative way. Using the Lagrangian multiplier method, $\mathcal{L}_1(\mathbf{G}, \mathbf{V})$ can be written as

$$\begin{aligned}\mathcal{L}_1{}'(\mathbf{G}, \mathbf{V}) = &tr\big((\mathbf{X} - \mathbf{XGV})(\mathbf{X} - \mathbf{XGV})^T\big)\\ &+ \gamma tr(\mathbf{VL}_s\mathbf{V}^T) + \varepsilon tr(\mathbf{V}^T\mathbf{QV}) + tr(\Phi\mathbf{G}^T)\\ &+ tr(\Psi\mathbf{V}^T)\end{aligned} \tag{12}$$

where $\Phi \in \mathbb{R}^{n\times k}$ and $\Psi \in \mathbb{R}^{k\times n}$ are the Lagrangian multipliers. The partial derivatives of $\mathcal{L}_1{}'(\mathbf{G}, \mathbf{V})$ with respect

to $\mathbf{G}$ or $\mathbf{V}$ are

$$\begin{cases} \dfrac{\partial \mathcal{L}_1'(\mathbf{G},\mathbf{V})}{\partial \mathbf{G}} = 2\mathbf{X}^T\mathbf{X}\mathbf{G}\mathbf{V}\mathbf{V}^T - 2\mathbf{X}^T\mathbf{X}\mathbf{V}^T + \Phi \\ \dfrac{\partial \mathcal{L}_1'(\mathbf{G},\mathbf{V})}{\partial \mathbf{V}} = 2\mathbf{G}^T\mathbf{X}^T\mathbf{X}\mathbf{G}\mathbf{V} - 2\mathbf{G}^T\mathbf{X}^T\mathbf{X} \\ \qquad\qquad + 2\gamma\mathbf{V}\mathbf{L} + 2\varepsilon\mathbf{Q}\mathbf{V} + \Psi. \end{cases} \tag{13}$$

Using the KKT conditions $\Phi \odot \mathbf{G} = \mathbf{0}$ and $\Psi \odot \mathbf{V} = \mathbf{0}$ (where "$\odot$" denotes the elementwise multiplication), and combining with formula (13), we obtain the following equation set:

$$\begin{cases} (2\mathbf{X}^T\mathbf{X}\mathbf{G}\mathbf{V}\mathbf{V}^T - 2\mathbf{X}^T\mathbf{X}\mathbf{V}^T) \odot \mathbf{G} = \mathbf{0} \\ \begin{pmatrix} 2\mathbf{G}^T\mathbf{X}^T\mathbf{X}\mathbf{G}\mathbf{V} - 2\mathbf{G}^T\mathbf{X}^T\mathbf{X} \\ +2\gamma\mathbf{V}\mathbf{L} + 2\varepsilon\mathbf{Q}\mathbf{V} \end{pmatrix} \odot \mathbf{V} = \mathbf{0}. \end{cases} \tag{14}$$

Then, $\mathbf{G}$ or $\mathbf{V}$ follow the multiplicative iterative updating rules:

$$\mathbf{G} \leftarrow \mathbf{G} \odot \frac{\mathbf{X}^T\mathbf{X}\mathbf{V}^T}{\mathbf{X}^T\mathbf{X}\mathbf{G}\mathbf{V}\mathbf{V}^T}, \tag{15}$$

$$\mathbf{V} \leftarrow \mathbf{V} \odot \frac{\mathbf{G}^T\mathbf{X}^T\mathbf{X} + \gamma\mathbf{V}\mathbf{S}}{\mathbf{G}^T\mathbf{X}^T\mathbf{X}\mathbf{G}\mathbf{V} + \gamma\mathbf{V}D + \varepsilon\mathbf{Q}\mathbf{V}} \tag{16}$$

where $\mathbf{A}/\mathbf{B}$ represents the elementwise division between matrixes $\mathbf{A}$ and $\mathbf{B}$.

Theorem 1: The loss function $\mathcal{L}_1(\mathbf{G},\mathbf{V})$ in (11) is nonincreasing under the updating rules in (15) and (16).

The Appendix A gives a detailed proof for the above theorem.

2) Fix $\mathbf{G}$, $\mathbf{V}$, $\mathbf{Y}_p$, and $\mathbf{S}$, and Update $\mathbf{W}$: When $\mathbf{G}$, $\mathbf{V}$, $\mathbf{Y}_p$, and $\mathbf{S}$ are fixed, items without $\mathbf{W}$ in $\mathcal{L}$ can be regarded as constant terms and they do not affect the optimization results. Therefore, when $\mathbf{G}$, $\mathbf{V}$, $\mathbf{Y}_p$, and $\mathbf{S}$ are fixed, model (10) has the same solution with the following formula:

$$\begin{aligned} \mathcal{L}_2 &= \left\|\mathbf{C}_n\left(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\right)\right\|_F^2 + \lambda\|\mathbf{W}\|_{2,1} \\ &= tr(\mathbf{C}_n\mathbf{X}^T\mathbf{W}\mathbf{W}^T\mathbf{X}\mathbf{C}_n) - 2tr(\mathbf{C}_n\mathbf{X}^T\mathbf{W}\mathbf{Y}_p^T\mathbf{C}_n) \\ &\quad + tr(\mathbf{C}_n\mathbf{Y}_p\mathbf{Y}_p^T\mathbf{C}_n) + \lambda\sum_{i=1}^{d}\sqrt{\sum_{j=1}^{c} w_{ij}^2}. \end{aligned} \tag{17}$$

Setting the derivative of $\mathcal{L}_2$ respect to $\mathbf{W}$ to be $\mathbf{0}$, we have

$$\frac{\partial \mathcal{L}_2}{\partial \mathbf{W}} = 2(\mathbf{X}\mathbf{C}_n\mathbf{X}^T\mathbf{W} + \lambda\Lambda\mathbf{W} - \mathbf{X}\mathbf{C}_n\mathbf{Y}_p) = \mathbf{0} \tag{18}$$

where Λ is a diagonal matrix and its diagonal elements can be written as follows:

$$\Lambda_{ii} = \frac{1}{2\sqrt{\sum_{j=1}^{c} w_{ij}^2}}, (i = 1, 2, \ldots, d).$$

To ensure that the denominator makes sense, a small enough positive constant Δ is added and Λ is transformed into $\Lambda' = diag(\Lambda'_{11}, \Lambda'_{22}, \ldots, \Lambda'_{dd})$ that the diagonal element can be written as follows:

$$\Lambda'_{ii} = \frac{1}{2\sqrt{\sum_{j=1}^{c} w_{ij}^2 + \Delta}}, (i = 1, 2, \ldots, d). \tag{19}$$

Algorithm 1 Alternative Iterative Algorithm to Solve Problem (17)

Require:

The Data matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$, $\mathbf{C_n} = \mathbf{I_n} - 1/n\mathbf{1}_{n\times n}$, the pseudo label matrix $\mathbf{Y}_p \in \mathbb{R}^{n\times c}$ and the hyper-parameters $\Delta > 0$, $\lambda > 0$.

Ensure:

The transformation matrix $\mathbf{W} \in \mathbb{R}^{n\times d}$.

Initialize $\Lambda' = \mathbf{I}_d$.

Repeat

1: Update $\mathbf{W} = (\mathbf{X}\mathbf{C}_n\mathbf{X}^T + \lambda\Lambda')^{-1}\mathbf{X}\mathbf{C}_n\mathbf{Y}_p$.

2: Update the diagonal matrix $\Lambda' \in \mathbb{R}^{d\times d}$ by

$$\Lambda'_{ii} = 1/2\sqrt{\sum_{j=1}^{c} w_{ij}^2 + \Delta}, (i = 1, 2, \cdots, d).$$

Until Convergence.

When replacing Λ' with Λ, (18) can be rewritten as follows:

$$\frac{\partial \mathcal{L}_2}{\partial \mathbf{W}} = 2(\mathbf{X}\mathbf{C}_n\mathbf{X}^T\mathbf{W} + \lambda\Lambda' - \mathbf{X}\mathbf{C}_n\mathbf{Y}_p) = \mathbf{0}. \tag{20}$$

According to (20), $\mathbf{W}$ can be expressed as the following formula:

$$\mathbf{W} = (\mathbf{X}\mathbf{C}_n\mathbf{X}^T + \lambda\Lambda')^{-1}\mathbf{X}\mathbf{C}_n\mathbf{Y}_p. \tag{21}$$

Considering that Λ' is corresponding to $\mathbf{W}$, it is unable to solve $\mathbf{W}$ from the (20) directly. Therefore, alternative iterative algorithm is utilized to find the optimal Λ' and $\mathbf{W}$. When $\mathbf{W}$ is fixed, Λ' can be obtained by (19), and conversely, $\mathbf{W}$ can be obtained by (21). The entire iterative process can be shown in Algorithm 1.

Theorem 2: The objective function in (17) is decreasing by the updating procedures in Algorithm 1.

Appendix B gives a detailed proof for the above theorem.

3) Fix $\mathbf{G}$, $\mathbf{V}$, $\mathbf{W}$, and $\mathbf{S}$, and Update $\mathbf{Y}_p$: When $\mathbf{G}$, $\mathbf{V}$, $\mathbf{W}$, and $\mathbf{S}$ are fixed, model (10) is equal to the following formula:

$$\begin{aligned} \mathcal{L}_3 &= \left\|\mathbf{C}_n\left(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\right)\right\|_F^2 + \alpha tr(\mathbf{Y}_p^T\mathbf{L}\mathbf{Y}_p) \\ &= tr\Big(\mathbf{Y}_p^T(\mathbf{C}_n + \alpha\mathbf{L})\mathbf{Y}_p\Big) - 2tr(\mathbf{Y}_p^T\mathbf{C}_n\mathbf{X}^T\mathbf{W}) \\ &\quad + tr(\mathbf{W}^T\mathbf{X}\mathbf{C}_n\mathbf{X}^T\mathbf{W}), \quad \text{s.t.} \quad \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c. \end{aligned} \tag{22}$$

Because $\mathbf{W}$ is fixed, $tr(\mathbf{W}^T\mathbf{X}\mathbf{C}_n\mathbf{X}^T\mathbf{W})$ is a constant term and problem (22) has the same solution with the following formula:

$$\begin{aligned} \mathcal{L}_3' &= tr\Big(\mathbf{Y}_p^T(\mathbf{C}_n + \alpha\mathbf{L})\mathbf{Y}_p\Big) - 2tr(\mathbf{Y}_p^T\mathbf{C}_n\mathbf{X}^T\mathbf{W}), \\ \text{s.t.} \quad & \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c. \end{aligned} \tag{23}$$

According to a previous literature [42], problem (23) is the standard form of quadratic on the stiefel manifold. Therefore, problem (23) could be solved by the generalized power iteration (GPI) method and complete proof procedure refers to [42]. In this article, the algorithm for $\mathbf{Y}_p$ is presented in Algorithm 2.

Algorithm 2 Alternative Iterative Algorithm to Solve Problem (23)

Require:

The Data matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$, $\mathbf{C_n} = \mathbf{I_n} - 1/n\mathbf{1}_{n\times n}$, the Laplacian matrix $\mathbf{L} \in \mathbb{R}^{n\times n}$, the transformation matrix $\mathbf{W} \in \mathbb{R}^{n\times d}$. and the hyper-parameters $\alpha > 0$.

Ensure:

The pseudo label matrix $\mathbf{Y}_p \in \mathbb{R}^{n\times c}$.

Initialize a random matrix $\mathbf{Y}_p \in \mathbb{R}^{n\times c}$ which satisfies $\mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c$.

Calculate

1: Calculate λ_A which is the maximum eigenvalue of matrix $\mathbf{A} = \mathbf{C}_n + \alpha\mathbf{L}$ calculated by power method and the matrix $\mathbf{A}' = \lambda_A\mathbf{I}_n - \mathbf{A}$.

2: Calculate the matrix $\mathbf{B} = \mathbf{C}_n\mathbf{X}^T\mathbf{W}$.

Repeat

1: Update $\mathbf{M} \leftarrow \mathbf{A}'\mathbf{Y}_p + 2\mathbf{B}$.

2: Decompose matrix $\mathbf{M} = \mathbf{U}_M\Sigma\mathbf{V}_M^T$ via *singular value decomposition method*.

3: Update $\mathbf{Y}_p = \mathbf{U}_M\mathbf{V}_M^T$.

Until Convergence.

4) Fix $\mathbf{G}$, $\mathbf{V}$, $\mathbf{W}$, *and* $\mathbf{Y}_p$, *and Update* $\mathbf{S}$*:* When $\mathbf{G}$, $\mathbf{V}$, $\mathbf{W}$, and $\mathbf{Y}_p$, the whole loss (10) is equal to the following formula:

$$\begin{aligned}\mathcal{L}_4 &= \alpha tr(\mathbf{Y}_p^T\mathbf{L}\mathbf{Y}_p) + \beta\sum_{i,j} s_{ij}\log s_{ij} + \gamma tr(\mathbf{V}\mathbf{L}\mathbf{V}^T),\\ &= \frac{1}{2}\alpha\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{r=1}^{c} s_{ij}(y_{ik}-y_{jk})^2 + \beta\sum_{i,j} s_{ij}\log s_{ij}\\ &\quad + \frac{1}{2}\gamma\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{q=1}^{k} s_{ij}(v_{qi}-v_{qj})^2 \quad \text{s.t.}\ \sum_{j=1}^{n} s_{ij} = 1, \mathbf{S}\geq\mathbf{0}.\end{aligned} \tag{24}$$

Using the Lagrangian multiplier method, $\mathcal{L}_4$ can be rewritten as follows:

$$\begin{aligned}\mathcal{L}_4' &= \frac{1}{2}\alpha\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{r=1}^{c} s_{ij}(y_{ir}-y_{jr})^2 + \beta\sum_{i,j} s_{ij}\log s_{ij}\\ &\quad + \frac{1}{2}\gamma\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{q=1}^{k} s_{ij}(v_{qi}-v_{qj})^2 + \sum_{i=1}^{n}\theta_i\left(\sum_{j=1}^{n} s_{ij} - 1\right)\\ &\quad + \sum_{i=1}^{n}\sum_{j=1}^{n}\pi_{ij}s_{ij}\end{aligned} \tag{25}$$

where $\Theta = [\,\theta_1\ \theta_2\ \cdots\ \theta_n\,]$ and $\Pi = [\pi_{ij}]_{n\times n}$ are Lagrangian multipliers. Using the KKT conditions, the optimal solution of (25) must satisfy the following formula:

$$\begin{cases}\dfrac{\partial L_4}{\partial s_{ij}} = \dfrac{\alpha}{2}\sum_{r=1}^{c}(y_{ir}-y_{jr})^2\\ \quad + \beta(\log s_{ij}+1) + \dfrac{\gamma}{2}\sum_{q=1}^{k}(v_{qi}-y_{qj})^2 = 0\\ s_{ij}\geq 0, \pi_{ij}\geq 0, \pi_{ij}s_{ij}\geq 0, \sum_{j=1}^{n} s_{ij} = 1.\end{cases} \tag{26}$$

Algorithm 3 Alternative Iterative Algorithm to Solve Problem (10)

Require:

The Data matrix $\mathbf{X} \in \mathbb{R}^{d\times n}$ and the hyper-parameters $\lambda > 0$, $\alpha > 0$, $\beta > 0$, $\gamma > 0$ and $\epsilon > 0$.

Ensure:

N selected features.

Initialize two random non-negative matrixes $\mathbf{G} \in \mathbb{R}^{n\times k}$ and $\mathbf{V} \in \mathbb{R}^{k\times n}$, a random matrix $\mathbf{Y}_p \in \mathbb{R}^{n\times c}$ which satisfies $\mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c$ and $\Lambda' = \mathbf{I}_d$.

Calculate

1: Calculate $\mathbf{S}$ by (27).

2: Calculate $\mathbf{D}$ by (7).

3: Calculate $\mathbf{L}$ by (6).

Repeat

1: Update $\mathbf{G}$ by (15).

2: Update $\mathbf{V}$ by (16).

3: Update $\mathbf{W}$ by algorithm 1.

4: Update $\mathbf{Y}_p$ by algorithm 2.

5: Calculate $\mathbf{S}$ by (27).

6: Calculate $\mathbf{D}$ and $\mathbf{L}$ by (7) and (6), respectively.

Until Convergence.

Sort all features according to $\sum_{j=1}^{d} w_{ij}^2$ in descending order and select the top N ranked features.

According to (26), we can obtain the optimal solution of (24), which can be written as follows:

$$s_{ij} = \frac{\exp\left\{\dfrac{\alpha\sum_{r=1}^{c}(y_{ir}-y_{jr})^2 + \gamma\sum_{q=1}^{k}(v_{qi}-v_{qj})^2}{2\beta}\right\}}{\sum_{j=1}^{n}\exp\left\{\dfrac{\alpha\sum_{r=1}^{c}(y_{ir}-y_{jr})^2 + \gamma\sum_{q=1}^{k}(v_{qi}-v_{qj})^2}{2\beta}\right\}}. \tag{27}$$

According to (27), we can know that when data $\mathbf{x}_i$ and $\mathbf{x}_j$ are similar (i.e., their pseudolabel and encoding vector are close, respectively), s_{ij} is close to 1, which satisfies the concept of similar matrix. Moreover, V and Y_p are interactive when updating the similarity matrix S. In other words, by manifold learning, the self-expression module and the pseudo-matrix learning are combined together in a single optimization process.

According to the aforementioned subalgorithm, the whole model (10) can be optimized by Algorithm 3. The optimal transformation matrix W, which projects the original data matrix into the pseudolabel matrix, is performed as the ranking scores to select the top representative features.

C. Computational Complexity Analysis

The computational complexity is a key indicator of algorithms. In this section, the computational complexity of the CNAFS is analyzed. According to Algorithm 3, the total optimization programme contains seven matrices that need to be updated (six steps). In each iteration, updating G needs

TABLE I
STATISTICS OF EIGHT BENCHMARK DATASETS

Dataset	# samples	# features	# classes
MNIST	3000	784	10
warpAR10P	130	2400	10
warpPIE10P	210	2420	10
COIL20	1440	1024	20
madelon	2600	500	2
JAFFE	213	676	10
Orlraws	100	10304	10
USPS	9298	256	10

$3dn^2 + 4n^2k + 2nk^2$ flmlt (a floating-point multiplication), updating V needs $4kdn + 3n^2k + 4nk^2$ flmlt, updating W needs $2dn^2 + 2nd^2 + dnc + 2dc$ flmlt, updating Y_p needs $n^2c + nc^2 + n^3$ flmlt, updating S needs $n^2c + n^2k$ flmlt, and updating D and L needs $2n$ flmlt. In many practical problems, $c < n, k < n, n < d, c \ll n$ are always true. Thus, the computational complexity of the proposed method is $O(nd^2 + n^2d + n^3)$.

IV. EXPERIMENT

A. Evaluation Benchmarks

A corpus with eight high-dimensionality benchmark datasets[1] are used to demonstrate the effectiveness of the proposed method. The datasets are chosen from different fields of study: handwritten digit recognition (MNIST[2] and USPS), face recognition (warpAR10P, warpPIE10P, JAFFE, and Orlraws), object image classification (COIL20), and highly nonlinear artificial data classification (madelon). These datasets are free from imbalance problems and Table I gives a summary of the statistics information for the datasets.

B. Evaluation Methodology

Following the same manner of previous methods, K-means clustering with a small feature subset of top importance scores is used to evaluate the effectiveness of FS. To measure the performance of clustering, two widely used evaluation metrics, that is, accuracy (ACC [43]) and the normalized mutual information (NMI [44]) are employed. The larger ACC and NMI represent better performance. These metrics are calculated by comparing the predicted label of each sample with the ground-truth labels provided in datasets [31]. ACC is defined by

$$\mathrm{ACC} = \frac{\sum_{i=1}^{n} \delta(\tau_i, \mathrm{map}(r_i))}{n} \tag{28}$$

where n represents the number of samples, τ_i denotes the ground-truth label of the ith sample, r_i denotes the corresponding predicted clustering label, $\delta(\cdot,\cdot)$ represents the Kronecker delta, and $\mathrm{map}(r_i)$ denotes the mapping function that finds the optimal match for the ith sample by the Kuhn–Munkres algorithm.

NMI measures the consistency between the predicted labels and the ground-truth labels. It is defined by

$$\mathrm{NMI} = \frac{2I(\tau_i, r_i)}{E(\tau_i) + E(r_i)} \tag{29}$$

where (τ_i, r_i) is mutual information between τ_i and r_i and $E(\cdot)$ returns the information entropy. Let n_i^r denote the sample number in the ith cluster generated by the clustering algorithm and n_i^τ denote the sample number in the ith cluster of the ground-truth label. Then, NMI is given by

$$\mathrm{NMI} = \frac{2\sum_{i=1}^{k}\sum_{j=1}^{k} n_{ij}\log\left(\frac{n-n_{ij}}{n_i^r n_j^i}\right)}{\sum_{i=1}^{k} n_i^r \log\frac{n_i^r}{n} + \sum_{i=1}^{k} n_i^\tau \log\frac{n_i^\tau}{n}} \tag{30}$$

where n_{ij} denotes the sample number of the intersection between τ_i and r_i.

C. Experimental Setup

To validate the effectiveness of the proposed approach in UFS, the proposed approach is compared with a baseline method that performs clustering with all the original features and eight other representative existing UFS methods. By comparing with these methods, we can evaluate the effectiveness of our method and its submodules from different perspectives. The involved methods are briefly introduced as follows.

1) *All-Fea:* All the original features are applied while clustering.
2) *Laplacian Score (LS) [25]:* LS employ the locality preserving power of each feature to evaluates the importance of feature.
3) Multi-Cluster Feature Selection (MCFS) [38]: MCFS selects the subset of features that is able to preserve the multicluster structure of the original data.
4) *Unsupervised Discriminative Feature Selection (UDFS) [39]:* UDFS proposes a joint model that combines the discriminative analysis with the $l_{2,1}$ minimization regularization to formulate the UFS problem.
5) *Uncorrelated Regression With Adaptive Graph for UFS (URAFS) [32]:* URAFS embeds the geometrical structure of data into the manifold learning and performs FS and manifold learning simultaneously with an uncorrelated regression model.
6) *Structure Preserving UFS (SPUFS) [31]:* SPUFS performs local structure learning by setting a structure preserved constraint in the self-expression model for maintaining the local manifold structure.
7) *Non-Negative Discriminative FS (NDFS) [45]:* NDFS combines the cluster label learning with FS for exploiting discriminative information from the raw data. Besides, a non-negative constraint is utilized in NDFS to obtain the clustering label more accurately.
8) *Subspace Clustering Guided CNMF (SC-CNMF) [41]:* SC-CNMF proposed a CNMF guided by the subspace

[1]Seven of the datasets downloaded from the ASU FS repository (http://featureselection.asu.edu/datasets.php).

[2]Downloaded from MNIST repository (http://yann.lecun.com/exdb/mnist/).

TABLE II
ACC (%± STD) STATISTICS FOR DIFFERENT METHODS ON EIGHT BENCHMARK DATASETS WHILE CLUSTERING WITH SELECTED FEATURES. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD

Dataset	MNIST	warpAR10P	warpPIE10P	madelon	COIL20	JAFFE	Orlraws	USPS
All-Fea	40.78 ± 2.18	23.84 ± 1.35	28.62 ± 3.26	50.30 ± 0.10	54.33 ± 3.67	75.12 ± 5.01	71.25 ± 8.89	65.35 ± 2.59
LS	41.86 ± 2.30	35.62 ± 3.58	39.76 ± 2.53	54.00 ± 4.51	61.00 ± 2.38	72.58 ± 5.42	65.35 ± 7.07	57.57 ± 2.76
MCFS	56.60 ± 2.86	34.08 ± 5.98	50.48 ± 5.73	54.23 ± 4.49	61.70 ± 4.05	76.71 ± 5.14	74.25 ± 6.63	62.98 ± 2.50
NDFS	43.31 ± 3.13	16.31 ± 1.41	24.71 ± 1.24	50.38 ± 0.01	61.60 ± 3.62	73.80 ± 5.52	76.90 ± 5.32	67.29 ± 2.35
UDFS	45.41 ± 3.37	38.08 ± 1.69	32.33 ± 2.48	54.74 ± 3.06	61.65 ± 3.74	70.61 ± 6.31	45.35 ± 2.87	46.64 ± 1.91
SPUFS	56.41 ± 1.82	35.46 ± 2.16	26.52 ± 1.63	56.04 ± 1.78	64.27 ± 2.32	75.02 ± 5.54	61.80 ± 4.18	70.55 ± 0.49
URAFS	57.27 ± 1.33	28.38 ± 3.61	29.57 ± 3.82	58.91 ± 2.78	61.22 ± 3.19	76.48 ± 3.85	68.30 ± 5.44	70.37 ± 2.43
SC-CNMF$_1$	53.68 ± 1.77	21.62 ± 1.68	31.38 ± 1.90	55.69 ± 4.33	66.31 ± 4.23	75.59 ± 4.62	77.24 ± 4.08	70.75 ± 1.35
SC-CNMF$_2$	52.91 ± 3.57	21.85 ± 1.93	31.76 ± 1.05	54.67 ± 3.87	64.17 ± 6.76	76.24 ± 4.69	76.74 ± 3.92	69.89 ± 1.14
CNAFS	**58.64 ± 1.60**	**44.23 ± 3.62**	**55.24 ± 2.42**	**60.32 ± 0.08**	**68.33 ± 3.86**	**79.81 ± 6.08**	**81.70 ± 3.92**	**71.65 ± 1.69**

TABLE III
NMI (%± STD) STATISTICS FOR DIFFERENT METHODS ON EIGHT BENCHMARK DATASETS WHILE CLUSTERING WITH SELECTED FEATURES. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD

Dataset	MNIST	warpAR10P	warpPIE10P	madelon	COIL20	JAFFE	Orlraws	USPS
All-Fea	34.85 ± 2.87	20.74 ± 3.91	25.90 ± 3.18	50.31 ± 0.10	71.33 ± 2.01	81.60 ± 2.80	79.25 ± 3.16	61.06 ± 1.48
LS	34.94 ± 1.79	39.56 ± 3.58	41.76 ± 2.53	52.99 ± 1.27	73.43 ± 2.01	81.47 ± 2.73	72.79 ± 2.83	55.47 ± 0.97
MCFS	45.79 ± 1.62	32.16 ± 5.98	56.70 ± 5.73	53.15 ± 1.33	76.07 ± 2.11	80.28 ± 2.89	**85.05 ± 3.32**	60.48 ± 0.97
NDFS	36.72 ± 2.89	12.21 ± 2.17	23.97 ± 2.64	51.42 ± 0.95	74.96 ± 1.86	74.05 ± 3.67	84.57 ± 2.81	**63.23 ± 1.33**
UDFS	37.10 ± 2.24	34.25 ± 1.69	32.05 ± 2.48	52.70 ± 0.88	71.78 ± 2.44	75.62 ± 2.72	49.75 ± 2.12	40.56 ± 0.79
SPUFS	46.42 ± 1.82	34.84 ± 2.17	19.34 ± 3.23	53.44 ± 0.40	76.73 ± 1.47	81.51 ± 3.62	58.17 ± 3.20	56.06 ± 1.10
URAFS	46.54 ± 1.33	23.82 ± 3.61	26.41 ± 2.95	57.54 ± 1.12	72.85 ± 1.56	79.25 ± 2.54	65.24 ±5.17	61.50 ± 1.34
SC-CNMF$_1$	43.88 ± 1.45	16.04 ± 3.51	30.51 ± 1.51	51.43 ± 1.22	76.64 ± 1.99	83.06 ± 1.84	79.02 ± 4.49	61.46 ± 1.74
SC-CNMF$_2$	42.60 ± 1.56	16.13 ± 5.25	30.89 ± 1.90	51.03 ± 1.22	77.45 ± 3.16	81.89 ± 4.46	78.68 ± 3.95	60.02 ± 1.15
CNAFS	**49.34 ± 1.60**	**44.51 ± 3.32**	**59.35 ± 3.67**	**59.28 ± 0.05**	**77.72 ± 2.81**	**85.05 ± 2.84**	79.21 ± 2.77	61.91 ± 1.45

clustering. Because of the different constraints, two different implementations are proposed, and the short of the two models is SC-CNMF$_1$ and SC-CNMF$_2$, respectively.

9) *Convex nonnegative matrix factorization with adaptive graph constraint (CNAFS):* The approach proposed in this article.

To evaluate the clustering performance of different unsupervised methods fairly, we use the author's implementation for SPUFS, URAFS, and UDFS and use Scikit-learn packages for the rest of the algorithms. The parameters for each comparing methods are chosen according to the author's literature. Specifically, for LS, URAFS, SPUFS, MCFS, UDFS, and our method, the size of nearest neighbors is set to 5 for all the datasets. The rest weight factors for different terms of the objective functions in the corresponding approach are tuned by searching from a grid of $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10^{1}, 10^{2}, 10^{3}\}$. The number of selected features is tuned from $\{20, 40, 60, \ldots, 200\}$ for all the datasets. The best clustering results from the optimal parameters are reported for all the algorithms. We employ the K-means algorithm to cluster the data points formed by the selected features to evaluate different methods. Considering that K-means algorithm is sensitive to initialization, K-means algorithm is repeated 20 times and the average clustering results are reported with standard deviation.

D. Experimental Results

The experimental results of different UFS methods on various datasets are presented in Table II in terms of ACC and Table III in terms of NMI. The best results are highlighted in bold. As can be seen from the two tables, the proposed CNAFS performs consistently better than other state-of-the-art methods in terms of both ACC and NMI, except that the NMI of CNAFS reaches a third place on Orlraws and a second place on USPS. Since CNAFS integrates self-expression into spectral analysis to remove noise or redundant information, it outperforms All-Fea, LS, and MCFS by providing a more efficient subset of the raw data for clustering. Our method also achieves better results than NDFS, UDFS, and URAFS for the main reason of employing convex NMF to the coefficient matrix of self-expression, which is more suitable for the data in real applications. Different from SPUFS, which regularizes the graph matrix in advance by predefined distance function measurement for the original data points, we propose an iteratively update strategy to optimize the similarity matrix according to the information entropy theory, which makes our method achieve better clustering results. Thus, the proposed approach is more capable while dealing with the noise in the original data and selecting the most representative features. Furthermore, MCFS, NDFS, UDFS, and URAFS use pesudo label matrix learning module to gain insights into the distribution of the true labels. SPUFS, SC-CNMF$_1$, and SC-CNMF$_2$ use self-expression module to explore the optimal weight coefficients for different feature subsets. While the proposed approach combines pseudolabel matrix learning module and self-expression module into a unified model whose results overwhelm all the aforementioned single-module methods.

To illustrate the intuitive effect of clustering, we visualize the data points and the clustering results by t-SNE with

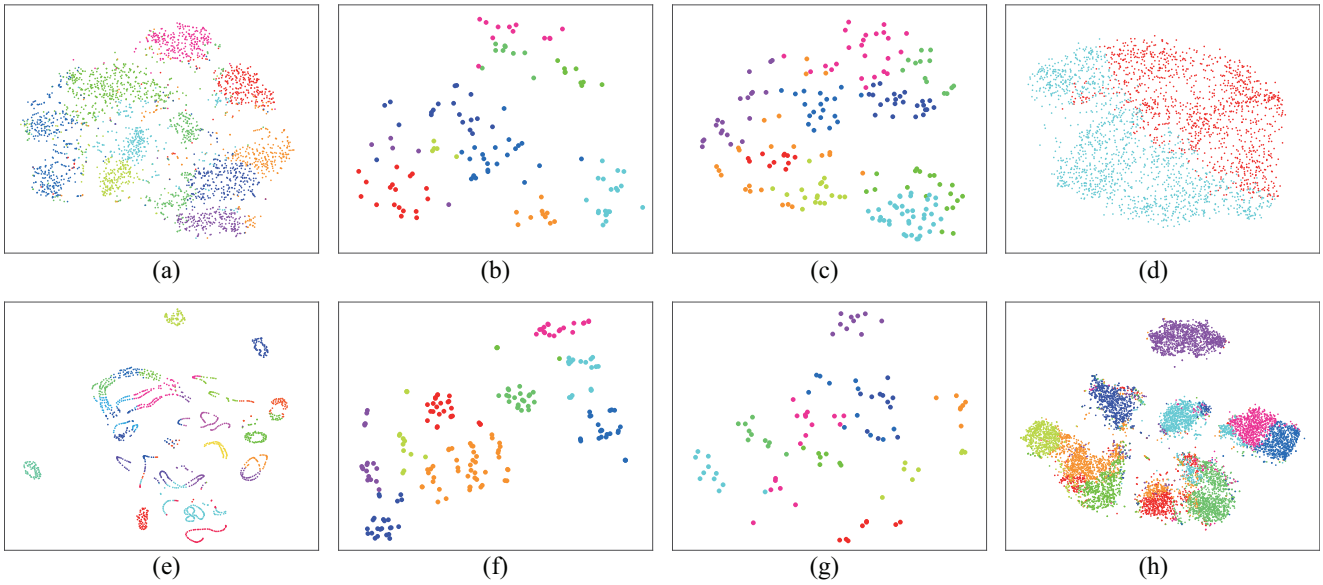

Fig. 1. Visualization of clustering results with *t*-SNE by CNAFS on the benchmark datasets. Best view in color. Points of the same color belongs to the same cluster. Clear boundaries lead to good clustering results while mixed boundaries denote areas of poor performance.

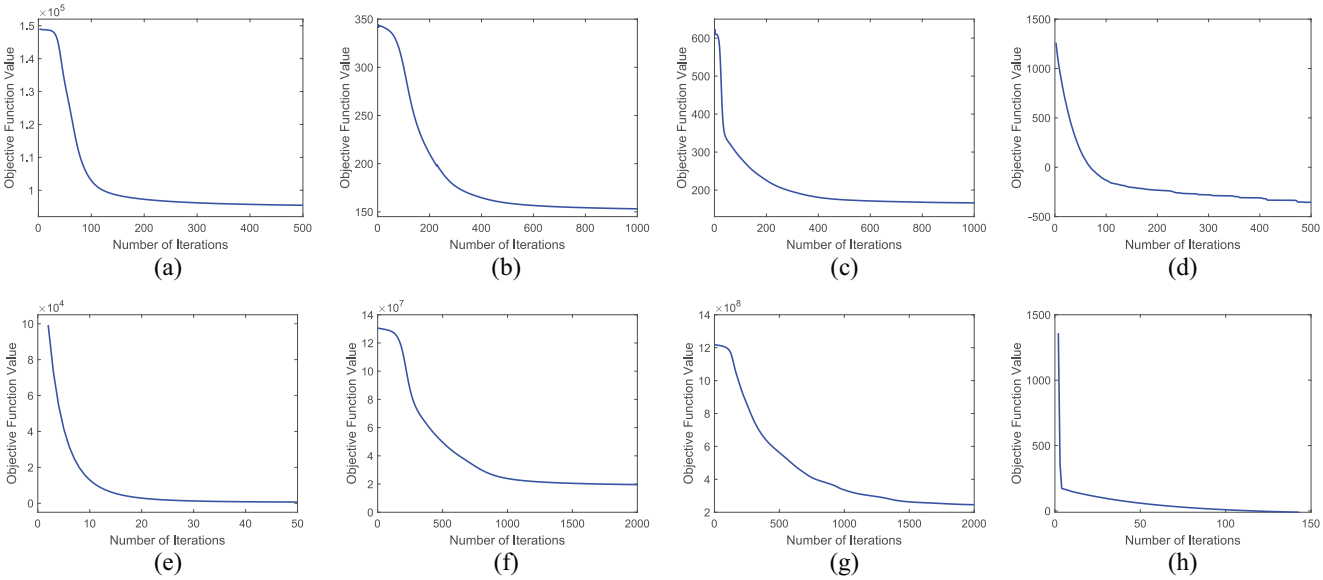

Fig. 2. Convergence curves of the proposed approach on the benchmark datasets.

selected features for different datasets in Fig. 1. It can be seen that visualized data points of MNIST, madelon, COIL20, JAFFE, Orlraws, and USPS are seperated into different clusters with relatively clear borders. In comparison, there is only a small number of data points for warpAP10p and warpPIE10p, and their clusters are blended with each other. This explains why the clustering result of warpAP10p and warpPIE10p is lower than that of other datasets. The convergence curves of the objective function values on different datasets during different iteration times are shown in Fig. 2. As can be seen from the figure, our method is convergent on all the datasets. The convergence is reached at different number of iteration for different datasets. Specifically, COIL20 converges with 50 iterations; MNIST and madelon converge with around 500 iterations; warpAR10p and warpPIE10p converge with 1000 iterations; JAFFE and Orlraws converges with the maximal 2000 iterations; and USPS converges with 100 iterations. Finally, the running times of LS, MCFS, UDFS, URAFS, SPUFS, and CNAFS on the dataset of COIL20 (the upper limits of iteration number are all set to 20) are compared in Table IV. The calculation is performed using an Intel Core i7-9700K CPU @ 3.60 GHz with 32.00-GB memory and 64-b Windows 10 operating system. On the other datasets, CNAFS needs more iterations to reach convergence than other methods as illustrated in Fig. 2. Although our method achieves better performance than the other state-of-the-art methods, the time-consuming optimization is a disadvantage of the proposed FS model.

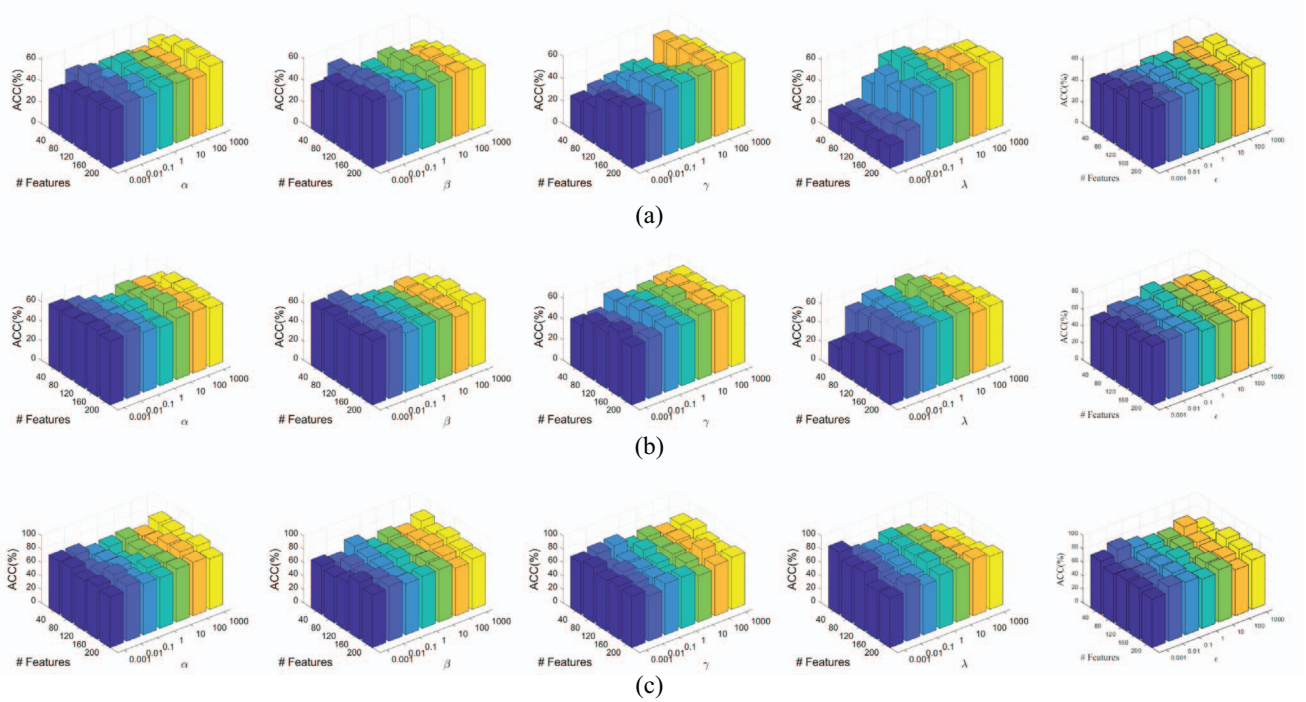

Fig. 3. Sensitivity of hyperparameters $\alpha, \beta, \gamma, \lambda$ and ϵ in terms of ACC on different datasets. The five subfigures in the same row demonstrate sensitivity of four parameters on the same dataset. The x-axis represents $\alpha, \beta, \gamma, \lambda$, and ϵ, respectively, while y-axis represents the number of the selected features, and z-axis represents the clustering accuracy. Specifically, the five subfigures in the first column apply setting: $\beta = 100$, $\gamma = 100$, $\lambda = 100$, and $\epsilon = 1$. The second column: $\alpha = 0.01$, $\gamma = 100$, $\lambda = 100$, and $\epsilon = 1$. The third column: $\alpha = 0.01$, $\beta = 100$, $\lambda = 100$, and $\epsilon = 1$. The fourth column: $\alpha = 0.01$, $\beta = 100$, $\gamma = 100$, and $\epsilon = 1$. The fifth column: $\alpha = 0.01$, $\beta = 100$, $\gamma = 100$, and $\lambda = 100$. Best view in color.

TABLE IV
STATISTICS OF THE RUNNING TIME IN SECONDS

Dataset	LS	MCFS	UDFS	URAFS	SPUFS	CNAFS
COIL20	0.10	12.75	9.19	20.58	1.42	50.85

E. Parameter Sensitivity and Ablation Study

First, the sensitiveness of the hyper parameters in our model is studied. The experimental results based on ACC and NMI metrics for all the datasets are reported in Fig. 3. The logarithms (base 10) of $\alpha, \beta, \gamma, \lambda$, and ϵ are taken. As we can see from Fig. 3, the clustering performance is comparatively sensitive to the parameters, which makes the parameter setting an open problem in FS.

The functions of different parts in the proposed model are studied by experiments of removing different parts. The effective combinations of each part in (10) are denoted as follows.

1) *Baseline (b):* The loss function composed of only the second and third items of (10) is set as the baseline, which is

$$\mathcal{L} = \left\|\mathbf{C}_n\left(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\right)\right\|_F^2 + \lambda\|\mathbf{W}\|_{2,1}, \quad \text{s.t.} \quad \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c. \tag{31}$$

2) $b+\alpha$*:* The loss function composed of only the baseline and the α weighted item, which is

$$\mathcal{L} = \left\|\mathbf{C}_n\left(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\right)\right\|_F^2 + \lambda\|\mathbf{W}\|_{2,1} + \alpha tr(\mathbf{Y}_p^T\mathbf{L}\mathbf{Y}_p), \quad \text{s.t.} \quad \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c. \tag{32}$$

3) $b+\alpha+\beta$*:* The loss function composed of only the baseline, α weighted item, and β weighted item, which is

$$\mathcal{L} = \left\|\mathbf{C}_n\left(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\right)\right\|_F^2 + \lambda\|\mathbf{W}\|_{2,1} + \alpha tr(\mathbf{Y}_p^T\mathbf{L}\mathbf{Y}_p) + \beta\sum_{i,j} s_{ij}\log s_{ij}, \quad \text{s.t.} \quad \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c, \sum_{j=1}^{n} s_{ij} = 1, \mathbf{S} \geq \mathbf{0}. \tag{33}$$

4) $b+\alpha+\beta$*+CNMF:* The loss function composed of only the baseline, the α, β, and γ weighted items, and the first item of (10), which is

$$\mathcal{L} = \|\mathbf{X} - \mathbf{X}\mathbf{G}\mathbf{V}\|_F^2 + \left\|\mathbf{C}_n\left(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\right)\right\|_F^2 + \lambda\|\mathbf{W}\|_{2,1} + \alpha tr(\mathbf{Y}_p^T\mathbf{L}\mathbf{Y}_p) + \beta\sum_{i,j} s_{ij}\log s_{ij} + \gamma tr(\mathbf{V}\mathbf{L}\mathbf{V}^T), \quad \text{s.t.} \quad \mathbf{Y}_p^T\mathbf{Y}_p = \mathbf{I}_c, \mathbf{G} \geq \mathbf{0}, \mathbf{V} \geq \mathbf{0}, \sum_{j=1}^{n} s_{ij} = 1, \mathbf{S} \geq \mathbf{0}. \tag{34}$$

5) *CNAFS:* The loss function of (10), which is also the proposed method.

As shown in Tables V and VI, the baseline model performs extremely bad in the madelon dataset in terms of ACC and better than $b + \alpha$ in most of the other datasets. The $b + \alpha + \beta$ and $b + \alpha + \beta$+CNMF models perform fairly well on the MNIST, madelon, and COIL20 datasets. However,

TABLE V
CLUSTERING RESULTS (ACC %± STD) OF DIFFERENT FS ALGORITHMS ON SIX BENCHMARK DATASETS. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD

Dataset	MNIST	warpAR10P	warpPIE10P	madelon	COIL20	JAFFE
Baseline (b)	34.39 ± 1.09	34.69 ± 2.85	38.38 ± 4.76	16.62 ± 2.51	62.47 ± 3.27	75.82 ± 6.26
b+α	30.33 ± 1.09	27.69 ± 4.85	33.05 ± 1.67	57.31 ± 0.03	59.62 ± 2.21	77.56 ± 4.43
b+α+β	56.04 ± 1.70	27.15 ± 3.12	35.14 ± 3.65	55.12 ± 0.08	64.10 ± 3.67	77.98 ± 7.72
b+α+β+CNMF	**59.24 ± 2.33**	26.54 ± 2.41	31.38 ± 3.79	**60.47 ± 0.09**	63.74 ± 1.96	74.08 ± 5.50
CNAFS	58.64 ± 1.60	**44.23 ± 3.62**	**55.24 ± 2.42**	60.32 ± 0.08	**68.33 ± 3.86**	**79.81 ± 6.08**

TABLE VI
CLUSTERING RESULTS (NMI %± STD) OF DIFFERENT FS ALGORITHMS ON SIX BENCHMARK DATASETS. THE BEST RESULTS ARE HIGHLIGHTED IN BOLD

Dataset	MNIST	warpAR10P	warpPIE10P	madelon	COIL20	JAFFE
Baseline (b)	25.02 ± 1.09	30.90 ± 2.85	38.39 ± 4.76	50.42 ± 0.51	73.62 ± 1.30	81.04 ± 4.45
b+α	19.27 ± 1.97	22.16 ± 2.78	29.03 ± 1.67	54.68 ± 0.01	72.41 ± 1.86	82.53 ± 3.38
b+α+β	45.68 ± 1.70	21.95 ± 3.12	34.41 ± 3.65	52.28 ± 0.02	74.83 ± 1.86	81.68 ± 5.54
b+α+β+CNMF	**50.22 ± 2.33**	19.69 ± 2.41	31.21 ± 3.79	**59.57 ± 0.05**	75.66 ± 1.39	78.59 ± 4.28
CNAFS	49.34 ± 1.60	**44.51 ± 3.32**	**59.35 ± 3.67**	59.28 ± 0.05	**77.72 ± 2.81**	**85.05 ± 2.84**

the proposed method, CNAFS, achieves the best performance with a large gap against the second best model on warpAR10p, warpPIE10P, COIL20, and JAFFE. Meanwhile, on MNIST and madelon, CNAFS is only slightly lower than the best models in terms of both ACC and NMI. By making compromise to make the majority of datasets to reach the best perfomrance, the final model of CNAFS balances different modules best. In conclusion, the different modules in the loss function of (10) cooperate best in the final CNAFS model.

V. CONCLUSION

We proposed a novel method based on convex non-negative matrix factorization with adaptive graph for UFS. Specially, self-expression and pseudolabel matrix learning are well embedded in one joint model to explore the data structure and the mapping relations between the data and the labels. Furthermore, two adaptive graph regularization terms are employed to keep the local structure of the raw data. The experimental results not only show that our CANFS obtains better performance on the six typical datasets than the state-of-the-art UFS methods but also show the effectiveness of the each elements of the proposed method.

APPENDIX A
PROOFS OF THEOREM 1

In order to proof Theorem 1, we follow the method proposed in [46], and we begin the definition of the auxiliary function.

Definition 1: $H(x, y)$ is an auxiliary function for $F(x)$ if the conditions

$$H(x, y) \geq F(x), H(x, x) = F(x) \tag{35}$$

are satisfied.

After the auxiliary function is defined, the following Lemma can be easily described.

Lemma 1: If $H(x, y)$ is an auxiliary function of $F(x)$, then $F(x)$ is nonincreasing under the update

$$x^{(t+1)} = \arg\min_{x} G\Big(x, x^{(t)}\Big). \tag{36}$$

Proof:

$$\begin{aligned} F\Big(x^{(t+1)}\Big) &\leq H\Big(x^{(t+1)}, x^{(t)}\Big) \\ &\leq H\Big(x^{(t)}, x^{(t)}\Big) = F\Big(x^{(t)}\Big). \end{aligned} \tag{37}$$

■

First, we proof that loss function $\mathcal{L}_1(\mathbf{G}, \mathbf{V})$ in (11) is non-increasing under the updating rules in (15) with a proper auxiliary function.

Considering any element G_{ab} in $\mathbf{G}$, we use $F_{ab}(G_{ab})$ to denote the part of $\mathcal{L}_1(\mathbf{G}, \mathbf{V})$, which is only relevant to G_{ab}. Setting the derivative of $F_{ab}(G_{ab})$ respect to G_{ab}, we have

$$\begin{aligned} F'_{ab}(G_{ab}) &= \left(\frac{\partial L_1}{\partial G_{ab}}\right) = \left(\frac{\partial L_1}{\partial G}\right)_{ab} \\ &= \left(2\mathbf{X}^T\mathbf{X}\mathbf{G}\mathbf{V}\mathbf{V}^T - 2\mathbf{X}^T\mathbf{X}\mathbf{V}^T\right)_{ab}. \end{aligned} \tag{38}$$

Setting the second-order derivative of $F_{ab}(G_{ab})$ respect to G_{ab}, we have

$$F''_{ab}(G_{ab}) = 2\left(\mathbf{X}^T\mathbf{X}\right)_{aa}\left(\mathbf{V}\mathbf{V}^T\right)_{bb}. \tag{39}$$

According to (39), the second-order derivative of $F_{ab}(G_{ab})$ is uncorrelated to G_{ab}. So, the higher order derivative of $F_{ab}(G_{ab})$ respect to G_{ab} equals to 0. Therefore, the Taylor series expansion of the $F_{ab}(G_{ab})$ is as follows:

$$\begin{aligned} F_{ab}\Big(G^{(t)}_{ab}\Big) = F_{ab}\Big(G^{(t)}_{ab}\Big) + F'_{ab}\Big(G^{(t)}_{ab}\Big)\Big(G_{ab} - G^{(t)}_{ab}\Big) \\ + \frac{F''_{ab}(G^{(t)}_{ab})}{2}\Big(G_{ab} - G^{(t)}_{ab}\Big)^2. \end{aligned} \tag{40}$$

Noting that the updating strategy in (15) is elementwise, therefore, we show that $F_{ab}(G^{(t)}_{ab})$ is decreasing with the updating strategy.

Lemma 2: Function

$$H\Big(G_{ab}, G_{ab}^{(t)}\Big) = F_{ab}\Big(G_{ab}^{(t)}\Big) + F'_{ab}\Big(G_{ab}^{(t)}\Big)\Big(G_{ab} - G_{ab}^{(t)}\Big) + \frac{\Big(\mathbf{X}^T\mathbf{X}\mathbf{G}^{(t)}\mathbf{V}\mathbf{V}^T\Big)_{ab}}{G_{ab}^{(t)}}\Big(G_{ab} - G_{ab}^{(t)}\Big)^2 \quad (41)$$

is an auxiliary function of $F_{ab}(G_{ab}^{(t)})$.

Proof: Obviously, $H(G_{ab}, G_{ab}) = F_{ab}(G_{ab})$. Therefore, we only need to proof $H(G_{ab}, G_{ab}^{(t)}) \geq F_{ab}(G_{ab})$. Comparing (40) and (41) , we should prove $(\mathbf{X}^T\mathbf{X}\mathbf{G}^{(t)}\mathbf{V}\mathbf{V}^T)_{ab} \geq \frac{1}{2}G_{ab}^{(t)}F''_{ab}(G_{ab}^{(t)})$. Considering that

$$\Big(\mathbf{X}^T\mathbf{X}\mathbf{G}^{(t)}\mathbf{V}\mathbf{V}^T\Big)_{ab} = \sum_{i=1}^{K}\sum_{j=1}^{n}(\mathbf{X}^T\mathbf{X})_{aj}G_{ij}^{(t)}(\mathbf{V}\mathbf{V}^T)_{bi} \geq G_{ab}^{(t)}(\mathbf{X}^T\mathbf{X})_{aa}(\mathbf{V}\mathbf{V}^T)_{bb} = \frac{1}{2}G_{ab}^{(t)}F''_{ab}(G_{ab}^{(t)}).$$

Therefore, $H(G_{ab}, G_{ab}^{(t)})$ is an auxiliary function for $F_{ab}(G_{ab})$. ■

When $G_{ab}^{(t)}$ is fixed, $H(G_{ab}, G_{ab}^{(t)})$ is a quadratic function about G_{ab} with opening upward. We can easily obtain the following formula:

$$\arg\min_{G_{ab}} H\Big(G_{ab}, G_{ab}^{(t)}\Big) = G_{ab}^{(t)}\frac{(\mathbf{X}^T\mathbf{X}\mathbf{V}^T)_{ab}}{(\mathbf{X}^T\mathbf{X}\mathbf{G}^{(t)}\mathbf{V}\mathbf{V}^T)_{ab}}.$$

According to Lemma 1, loss function $\mathcal{L}_1(\mathbf{G}, \mathbf{V})$ in (11) is decreasing under the updating rules in (15).

Second, we can easily proof that the loss function $\mathcal{L}_1(\mathbf{G}, \mathbf{V})$ in (11) is nonincreasing under the updating rules in (16) by using the similar idea, and the proving process is omitted.

APPENDIX B
PROOFS OF THEOREM 2

In order to prove Theorem 1, we give the following lemma at first.

Lemma 3: For any positive real numbers x and y, they satisfy the following inequality:

$$\sqrt{x} - \frac{x}{2\sqrt{y}} \leq \sqrt{y} - \frac{y}{2\sqrt{y}}. \quad (42)$$

Proof: Note that the solution of (20) is equal to the solution of the following problem:

$$\min_{\mathbf{W}} \big\|\mathbf{C}_n\big(\mathbf{X}^T\mathbf{W} - \mathbf{Y}_p\big)\big\|_F^2 + \lambda tr\big(\mathbf{W}^T\Lambda'\mathbf{W}\big). \quad (43)$$

Denote the tth and $(t+1)$th iteration of $\mathbf{W}$ with $\mathbf{W}^t$ and $\mathbf{W}^{(t+1)}$, respectively, it must hold the following inequality:

$$\Big\|\mathbf{C}_n\Big(\mathbf{X}^T\mathbf{W}^{(t+1)} - \mathbf{Y}_p\Big)\Big\|_F^2 + \lambda tr\Big(\big(\mathbf{W}^{(t+1)}\big)^T\Lambda'\mathbf{W}^{(t+1)}\Big) \leq \Big\|\mathbf{C}_n\Big(\mathbf{X}^T\mathbf{W}^{(t)} - \mathbf{Y}_p\Big)\Big\|_F^2 + \lambda tr\Big(\big(\mathbf{W}^{(t)}\big)^T\Lambda'\mathbf{W}^{(t)}\Big). \quad (44)$$

It can be also written as follows:

$$\Big\|\mathbf{C}_n\Big(\mathbf{X}^T\mathbf{W}^{(t+1)} - \mathbf{Y}_p\Big)\Big\|_F^2 + \lambda\sum_{i=1}^{n}\frac{\sum_{j=1}^{c}\Big(w_{ij}^{(t+1)}\Big)^2 + \Delta}{2\sqrt{\sum_{j=1}^{c}\Big(w_{ij}^{(t)}\Big)^2 + \Delta}} \leq \Big\|\mathbf{C}_n\Big(\mathbf{X}^T\mathbf{W}^{(t)} - \mathbf{Y}_p\Big)\Big\|_F^2 + \lambda\sum_{i=1}^{n}\frac{\sum_{j=1}^{c}\Big(w_{ij}^{(t)}\Big)^2 + \Delta}{2\sqrt{\sum_{j=1}^{c}\Big(w_{ij}^{(t)}\Big)^2 + \Delta}}. \quad (45)$$

According to Lemma 3, we have the following inequality:

$$\lambda\sum_{i=1}^{n}\sqrt{\sum_{j=1}^{c}\Big(w_{ij}^{(t+1)}\Big)^2 + \Delta} - \lambda\sum_{i=1}^{n}\frac{\sum_{j=1}^{c}\Big(w_{ij}^{(t+1)}\Big)^2 + \Delta}{2\sqrt{\sum_{j=1}^{c}\Big(w_{ij}^{(t)}\Big)^2 + \Delta}} \leq \lambda\sum_{i=1}^{n}\sqrt{\sum_{j=1}^{c}\Big(w_{ij}^{(t)}\Big)^2 + \Delta} - \lambda\sum_{i=1}^{n}\frac{\sum_{j=1}^{c}\Big(w_{ij}^{(t)}\Big)^2 + \Delta}{2\sqrt{\sum_{j=1}^{c}\Big(w_{ij}^{(t)}\Big)^2 + \Delta}}. \quad (46)$$

Combining inequalities (45) and (46), we can obtain the following inequality:

$$\Big\|\mathbf{C}_n\Big(\mathbf{X}^T\mathbf{W}^{(t+1)} - \mathbf{Y}_p\Big)\Big\|_F^2 + \lambda\Big\|\mathbf{W}^{(t+1)}\Big\|_{2,1} \leq \Big\|\mathbf{C}_n\Big(\mathbf{X}^T\mathbf{W}^{(t)} - \mathbf{Y}_p\Big)\Big\|_F^2 + \lambda\Big\|\mathbf{W}^{(t)}\Big\|_{2,1}. \quad (47)$$

Then, it shows that the objective function in (17) is decreasing by the updating procedures in Algorithm 1. ■

REFERENCES

[1] L. Yu and H. Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *Proc. 20th Int. Conf. Mach. Learn. (ICML)*, 2003, pp. 856–863.

[2] E. P. Xing, M. I. Jordan, and R. M. Karp, "Feature selection for high-dimensional genomic microarray data," in *Proc. 18th Int. Conf. Mach. Learn. (ICML)*, 2001, pp. 601–608.

[3] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: A review," *SIGKDD Explorations Newslett.*, vol. 6, no. 1, pp. 90–105, 2004.

[4] F. Luo, L. Zhang, B. Du, and L. Zhang, "Dimensionality reduction with enhanced hybrid-graph discriminant learning for hyperspectral image classification," *IEEE Trans. Geosci. Remote. Sens.*, vol. 58, no. 8, pp. 5336–5353, Aug. 2020.

[5] F. Luo, L. Zhang, X. Zhou, T. Guo, Y. Cheng, and T. Yin, "Sparse-adaptive hypergraph discriminant analysis for hyperspectral image classification," *IEEE Geosci. Remote. Sens. Lett.*, vol. 17, no. 6, pp. 1082–1086, Jun. 2020.

[6] F. Luo, B. Du, L. Zhang, L. Zhang, and D. Tao, "Feature learning using spatial-spectral hypergraph discriminant analysis for hyperspectral image," *IEEE Trans. Cybern.*, vol. 49, no. 7, pp. 2406–2419, Jul. 2019.

[7] J. Gui, Z. Sun, S. Ji, D. Tao, and T. Tan, "Feature selection based on structured sparsity: A comprehensive study," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 7, pp. 1490–1507, Jul. 2017.

[8] C. Hou, F. Nie, X. Li, D. Yi, and Y. Wu, "Joint embedding learning and sparse regression: A framework for unsupervised feature selection," *IEEE Trans. Cybern.*, vol. 44, no. 6, pp. 793–804, Jun. 2014.

[9] R. Zhang, F. Nie, and X. Li, "Self-weighted supervised discriminative feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3913–3918, Aug. 2018.

[10] R. Zhang, F. Nie, Y. Wang, and X. Li, "Unsupervised feature selection via adaptive multimeasure fusion," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 9, pp. 2886–2892, Sep. 2019.

[11] S. Alelyani, J. Tang, and H. Liu, "Feature selection for clustering: A review," in *Data Clustering: Algorithms and Applications*. Boca Raton, FL, USA: CRC Press, 2013, pp. 29–60.

[12] X. Li, M. Chen, F. Nie, and Q. Wang, "A multiview-based parameter free framework for group detection," in *Proc. 31st AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, 2017, pp. 4147–4153.
[13] K. Zhan, X. Chang, J. Guan, L. Chen, Z. Ma, and Y. Yang, "Adaptive structure discovery for multimedia analysis using multiple features," *IEEE Trans. Cybern.*, vol. 49, no. 5, pp. 1826–1834, May 2019.
[14] X. Li, M. Chen, F. Nie, and Q. Wang, "Locality adaptive discriminant analysis," in *Proc. 26th Int. Joint Conf. Artif. Intell. (IJCAI)* Melbourne, VIC, Australia, Aug. 2017, pp. 2201–2207.
[15] C. Lu, H. Min, J. Gui, L. Zhu, and Y. Lei, "Face recognition via weighted sparse representation," *J. Vis. Commun. Image Represent.*, vol. 24, no. 2, pp. 111–116, 2013.
[16] J.-X. Mi, D. Lei, and J. Gui, "A novel method for recognizing face with partial occlusion via sparse representation," *Optik*, vol. 124, no. 24, pp. 6786–6789, 2013.
[17] K. Zhan, C. Zhang, J. Guan, and J. Wang, "Graph learning for multiview clustering," *IEEE Trans. Cybern.*, vol. 48, no. 10, pp. 2887–2895, Oct. 2018.
[18] K. Zhan, C. Niu, C. Chen, F. Nie, C. Zhang, and Y. Yang, "Graph structure fusion for multiview clustering," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 10, pp. 1984–1993, Oct. 2019.
[19] K. Zhan, F. Nie, J. Wang, and Y. Yang, "Multiview consensus graph clustering," *IEEE Trans. Image Process.*, vol. 28, no. 3, pp. 1261–1270, Mar. 2019.
[20] J. M. Sotoca and F. Pla, "Supervised feature selection by clustering using conditional mutual information-based distances," *Pattern Recognit.*, vol. 43, no. 6, pp. 2068–2081, 2010.
[21] Z. Xu, I. King, M. R. Lyu, and R. Jin, "Discriminative semi-supervised feature selection via manifold regularization," *IEEE Trans. Neural Netw.*, vol. 21, no. 7, pp. 1033–1047, Jul. 2010.
[22] X. Chang, F. Nie, Y. Yang, and H. Huang, "A convex formulation for semi-supervised multi-label feature selection," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 1171–1177.
[23] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. New York, NY, USA: Wiley, 2012.
[24] Q. Gu, Z. Li, and J. Han, "Generalized fisher score for feature selection," in *Proc. 27th Conf. Uncertainty Artif. Intell.*, 2011, pp. 266–273.
[25] X. He, D. Cai, and P. Niyogi, "Laplacian score for feature selection," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2005, pp. 507–514.
[26] L. Wolf and A. Shashua, "Feature selection for unsupervised and supervised inference: The emergence of sparsity in a weight-based approach," *J. Mach. Learn. Res.*, vol. 6, pp. 1855–1887, Dec. 2005.
[27] S. Maldonado and R. Weber, "A wrapper method for feature selection using support vector machines," *Inf. Sci.*, vol. 179, no. 13, pp. 2208–2217, 2009.
[28] P. Bermejo, J. A. Gámez, and J. M. Puerta, "Speeding up incremental wrapper feature subset selection with naive bayes classifier," *Knowl. Based Syst.*, vol. 55, pp. 140–147, Jan. 2014.
[29] L. Cheng, Y. Wang, X. Liu, and B. Li, "Outlier detection ensemble with embedded feature selection," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 3503–3512.
[30] X. Zhu, X. Li, S. Zhang, C. Ju, and X. Wu, "Robust joint graph sparse coding for unsupervised spectral feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 6, pp. 1263–1275, Jun. 2017.
[31] Q. Lu, X. Li, and Y. Dong, "Structure preserving unsupervised feature selection," *Neurocomputing*, vol. 301, pp. 36–45, Aug. 2018.
[32] X. Li, H. Zhang, R. Zhang, Y. Liu, and F. Nie, "Generalized uncorrelated regression with adaptive graph for unsupervised feature selection," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 5, pp. 1587–1595, May 2019.
[33] H. Peng, F. Long, and C. H. Q. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005.
[34] M. Masaeli, G. Fung, and J. G. Dy, "From transformation-based dimensionality reduction to feature selection," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 751–758.
[35] S. Tabakhi, P. Moradi, and F. Akhlaghian, "An unsupervised feature selection algorithm based on ant colony optimization," *Eng. Appl. Artif. Intell.*, vol. 32, pp. 112–123, Jun. 2014.
[36] J. G. Dy and C. E. Brodley, "Feature selection for unsupervised learning," *J. Mach. Learn. Res.*, vol. 5, pp. 845–889, Dec. 2004.
[37] Z. Zhao and H. Liu, "Spectral feature selection for supervised and unsupervised learning," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, 2007, pp. 1151–1157.
[38] D. Cai, C. Zhang, and X. He, "Unsupervised feature selection for multi-cluster data," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2010, pp. 333–342.
[39] Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou, "$l_{2,1}$-norm regularized discriminative feature selection for unsupervised learning," in *Proc. 22nd Int. Joint Conf. Artif. Intell. (IJCAI)*, 2011, pp. 1589–1594.
[40] M. Qian and C. Zhai, "Robust unsupervised feature selection," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2013, pp. 1621–1627.
[41] G. Cui, X. Li, and Y. Dong, "Subspace clustering guided convex non-negative matrix factorization," *Neurocomputing*, vol. 292, pp. 38–48, May 2018.
[42] F. Nie, R. Zhang, and X. Li, "A generalized power iteration method for solving quadratic problem on the stiefel manifold," *Sci. China Inf. Sci.*, vol. 60, no. 11, pp. 1–10, 2017.
[43] X. Cao, C. Zhang, H. Fu, S. Liu, and H. Zhang, "Diversity-induced multi-view subspace clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, 2015, pp. 586–594.
[44] A. Strehl and J. Ghosh, "Cluster ensembles—A knowledge reuse framework for combining multiple partitions," *J. Mach. Learn. Res.*, vol. 3, pp. 583–617, Mar. 2003.
[45] Z. Li, Y. Yang, J. Liu, X. Zhou, and H. Lu, "Unsupervised feature selection using nonnegative spectral analysis," in *Proc. 26th AAAI Conf. Artif. Intell.*, 2012, pp. 1026–1032.
[46] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2000, pp. 556–562.

Aihong Yuan received the Ph.D. degree from the Center for Optical Imagery Analysis and Learning, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an, China, in 2019.

He is a Lecturer with the Computer Science Department, Northwest A&F University, Xianyang, China. His research interests include machine learning, image/video content understanding, and deep learning.

Mengbo You received the Ph.D. degree from Iwate University, Morioka, Japan, in 2018.

He is a Lecturer with the Computer Science Department, Northwest A&F University, Xianyang, China. His research interests include machine learning, image processing, object detection, and deep learning.

Dongjian He received the B.E., M.E., and D.E., degrees in agricultural engineering from Northwest A&F University, Xianyang, China, in 1982, 1985, and 1998, respectively.

He was a Lecturer with the College of Mechanical and Electronic Engineering, Northwest A&F University, from 1987 to 1992, where he was an Associate Professor from 1992 to 1999. He is currently a Professor with the College of Mechanical and Electronic Engineering, Northwest A&F University. His research interests include computer graphics, image analysis, and machine vision.

Dr. He is a Member of the China Computer Federation, the Chairman of the Shaanxi Society of Image and Graphics, the Vice Chairman of the Electrical Information and Automation Committee of CSAE, and a Member of the Council of the Chinese Society for Agricultural Machinery.

Xuelong Li (Fellow, IEEE) is a Full Professor with the School of Computer Science and Center for Optical Imagery Analysis and Learning, Northwestern Polytechnical University, Xi'an, China.